Proceedings of the 1999 IEEE
International Symposium on Assembly and Task Planning
Porto, Portugal - July 1999

MULTI-AGENT SIMULATION FOR BALANCING OF ASSEMBLY LINES

Isabel C. Praça & Carlos Ramos

Departamento de Informática
Instituto Superior de Engenharia do Porto
Rua São Tomé
4200 Porto, Portugal
Tel: + 351 - 2 - 8340500 email: { isp / csr }@dei.isep.ipp.pt

Abstract

One of the main applications of simulation is in manufacturing systems. The complexity of this kind of systems is related to the number of decisions that must be taken in shorter time. The main difficulty is to predict the effects of decisions on the overall system performance. In this context simulation becomes an important tool by means of which it is possible to predict the future performance of the system, and so compare and analyse different decisions. The actual discrete-event simulation scenario reveals a systematic demand for more sophisticated and powerful computational tools in order to fulfil the increasing of problem solving complexity. Computer network technology provides the computational power needed. The combination of Distributed Simulation with that of Multi-Agent Systems presents a very interesting perspective to design and develop such kind of environments. We propose an architecture based on Multi-Agent Simulation to help solving problems in balancing and distributing the human resources in Manual Assembly Lines, and in the future we hope to extend the model to other kinds of manufacturing systems.

Keywords

Simulation, Manual Assembly Lines, Balancing, Multi-Agent Systems.

Introduction

Applications of simulation in the manufacturing area cover a broad variety of issues. This kind of systems deals with a lot of problems and there are several decisions that must be taken in shorter time in order to meet customers demand, to keep the production facilities work efficiently and to solve random problems that occur, such as equipment breakdown.

In manufacturing systems organised as flow lines, one of the most important issue is the line balancing problem, that is a NP-hard problem, which is usually solved by the application of heuristic rules. Concerning manual flow lines the problems are even more complex because of the variability associated with human behaviour.

The simulation emerges in this context as a very important framework to analyse and compare different line configurations. It becomes an important component in a decision making system to help on the manual lines balancing problem, since it can give the user information about the future performance of alternative configurations for the line.

The analysis of a manufacturing system organised as a set of manual flow lines has served as starting point to the development of an information system (SimBa) which gives the user the possibility of balancing a line with three different heuristic rules, and then compare the configurations obtained by simulation. This system is described, and its limitations are pointed out.

In this work we also present the specification of a Multi-Agent architecture for improving the limitations of the SimBa system. With this architecture the simulation of more than one line, the simulation under transient periods and the improvement in the correct distribution of the operators according to their skills becomes possible. The future implementation of the architecture is being studied, and the utilisation of the Swarm Toolkit [7] seems very promising.

Advantages and Applications of Simulation in Manufacturing Systems

Simulation can be used for dealing intractable problems, those that are too difficult or complex to solve mathematically. Analytical solutions to some issues may not be achievable. Trail and error simulations may, however, lead to a near optimal solution.

Simulation can be a valuable training tool. Through it we can gain a better understanding of the system's performance than we could achieve by merely solving an equation for the optimal value. An optimisation formula may apply only under *steady state conditions*, but because the actual environment for a system may vary over time, it is often the *non steady state conditions* that bear investigating. With simulation it is possible to study a system under *non steady state*.

Simulation allows the study of a system under controlled conditions. A model is not subject to the *Hawthorne effect*, where people's behaviour is modified because they are being studied. The investigator can select the variables to be changed and the extent to which they are changed. Results can be obtained for various combinations of environmental and internal operation policies that might not occur in the real environment within a reasonable time. Simulation can then project conditions that might occur in the future so that a company can be prepared for them. So, with simulation it is possible to do a sensitive analysis and evaluate the future performance of the system.

Simulation can expand or compress time to provide a more detailed review of certain events.

Simulation can be less expensive and involve less risk than actual experimentation. It is often less expensive to change a replica than to change some aspects of the real world. If some alternative led to a failure or serious damage of the real system, it would be expensive or even impossible to restore the original conditions so that another alternative could be tried.

Also, simulation does not optimise; it just shows what the model says would result if a certain alternative were tried

0-7803-5704-3/99/$10.00©1999 IEEE

under a particular set of conditions. Even with an accurate or valid model there is a danger that the investigator will not try the alternatives that would provide the greatest benefit.

In manufacturing systems applications of simulation are numerous. Some of the well-know applications include the following:

- Facilities design. These applications are related to the size of facilities or the number of servers required;
- Aggregate planning. Where an operation's aggregate capacity must be determined, simulation models are used to find cost of alternative plans;
- Scheduling. Simulation as been used in evaluation of alternative scheduling rules. In the scheduling of jobs in a machine shop different and alternative dispatching rules can be evaluated by simulation;
- Inventory. Many inventory models are evaluated by simulating the effect of ordering rules. In complex inventory situations, it is often necessary to simulate proposed rules before they are put into effect to determine the impact on customer service and cost;
- Materials requirements planning. When materials requirement planning (MRP) is used to plan and control manufacturing, simulation is used to evaluate proposed changes in the manufacturing plan before the changes are put into effect. As a result, "what if" questions can be asked by a management before decisions are made.

Simulation is often the tool for the analysis of queuing situations because of the complexity of many queuing problems and because analysts frequently want data for non-steady conditions. A wide variety of situations can be formulated as queuing problems. Such is the case of manufacturing flow lines, where each workstation is represented as having one or more servers and an associated waiting queue with an input buffer, so the complete line is like a set of queues that feed each other.

Manual Flow Lines Balancing and Simulation

The simulation of flow line production systems is of discrete event type [6], since what is determinant is that the state of the system changes only when some events occur. Some of those events are:

- the end of batch (or item) processing in a work station;
- breakdown of a machine;
- lack of raw materials;
- unavailable operator.

In manual manufacturing lines the variability of the human behaviour influences the processing times, and so the operations duration change in time. The variability is due to some factors such as:

- learning;
- labour turnover;
- absenteeism;
- workers motivation;
- and, the workers physical and psychological state, which influences all the others.

It can be modelled by means of an exponential learning curve [11], and random factors like absenteeism can be modelled as random events.

The analysis of a manufacturing system organised as a set of manual flow lines, and the difficulties of the managers in taking decisions has served as motivation to the development of a System for Line Balancing and Simulation (SimBa).

In that manufacturing system the simulation tool is very important to:

- compare the line configurations based on the criteria specified;
- test the sensitivity to some random events like absenteeism and breakdown of equipment.

The most important characteristics of the developed system, called SimBa [8] [9], are the implementation of three heuristic rules, selected to give different line configurations which can after be compared by means of simulation. The rules were selected according to the study developed by Boctor [1] which analyse their performance, and the simplicity of the rules, which is an important factor when considering future improvements of the rules to adapt then to the situation in question.

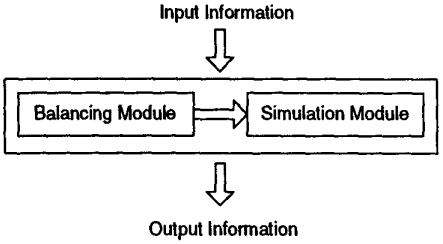

Fig.1 Architecture of SimBa

Line Balancing Module

The Line Balancing Module implements three heuristic rules which allow the user to obtain different line configurations.

Input Information

- Manufacturing characteristics of the product: number, standard durations and precedence restrictions of operations;
- Number of items to produce and lot size characteristics;

Heuristic rules

- Largest Candidate Rule;
- Ranked Positional Weights;
- Multiple-rule Heuristic by Boctor.

Output Information

- Line configuration;

Line Simulation Module

The simulation module is used to evaluate the future performance of the system and compare different line configurations based on the criteria mentioned, obtained by the implemented rules or other suggested by the user.

It is possible to simulate the functioning of lines with the standard durations of the operations and also taking into account the operators learning curves, which allows an

analysis that reflects the variability of human performance. Using a simulation considering the learning curves of the operators allows the user to test different allocations in order to analyse which is the best place for each operator.

Input Information

- Graph of operations (precedence restrictions);
- Line configuration (number of workstations and which operations are done in each one);
- Learning curves of workers (if we want to simulate a manual line).

It is a discrete event simulation based on a queuing model, where each work station is represented as having one or more servers and an associated waiting queue that represents the input buffer. When simulating the line taking into account the operators performance, the continuous state variables that represent the human work, modelled by an exponential learning curve, it becomes a combined discrete continuous simulation where the continuous variables originate a discrete event when they reach some thresholds.

Output Information:

- levels of stocks between work stations;
- work stations utilisation times;
- time to complete the production of all the items.

With this information it is possible to compare different line configurations. A line configuration with lower levels of stocks between the work stations, high utilisation times and fewer time to complete the production of all items is the most suitable solution. The information concerning the levels of stocks in the input buffer of each work station allows the detection of bottleneck stations, by the high level of stock in the input buffer and high utilisation time.

Based on the output information of the SimBa system, the user can support the decision of choosing the line configuration that better fits the production and do some adjustments in order to smooth the alterations due to the frequent change of product being manufactured.

The SimBa system was developed according to an object oriented methodology, specifically the "Object Modelling Technique" by Rumbaugh [10]. It was developed in C++, and the Line Simulation Module was also supported by Sim++ libraries of the SimPack package developed by Prof. Fishwick team at the University of Florida [3].

The application of the system developed in the real manufacturing system as served to validate the results obtained and to achieve the conclusion that a system like this with an user friendly interface can be a very helpful framework to the managers for the line balancing. In [8] the system and the results obtained are described.

However there are some characteristics that are not implemented in SimBa, such as:

- the simulation of multi-model and mixed-model manufacturing lines;
- the simulation of the line under transient periods. In SimBa the only possibility is to analyse another line configuration and not the period in which there are two different models being made in the same line, due to the introduction of another model in the line while it is still finishing the first model. This is very important in systems where the line configuration is frequently changed due to the small production batches;
- the possibility of simulating a set of lines, and not just one line;
- the distribution of human resources among the different workstations of the line, according to the experience and skills of the workers;
- the hierarchical structure of manufacturing systems with several departments, some of them organised as manufacturing lines. This improvement is very important, because all the departments should work together as a whole for the objective of the system.

It is not possible to provide these characteristics using SimBa system, because it is not very flexible, it just covers the simulation of a single line, according to some heuristic rules, manufacturing a single product, with some well predefined manufacturing sequence.

Because of SimBa limitations and because the implementation of the referred characteristics increases greatly the simulation, we are working on developing an architecture based on Multi-Agent Systems that is powerful and more flexible. In the future we hope to extend the architecture to other kinds of situations and other types of manufacturing systems.

Multi-Agent Systems

Different frames of mind may be observed in humans when faced with a problem. An entering upon a problem's scenario, looking for clues that may help in the problem's solution, it is a very common one. Such a strategy may eventually provide the basis to develop an effective plan to attack efficiently a problem and solve it. Reacting to stimulus generated from environment changes is another typical form of human attitudes. This allows for humans to choose, for a given event, the best way to react and consequently to apply a better strategy. The difficulty to establish a comparative model to evaluate properly these two approaches is significative. A verdict to apply one or the other may be induced by the problem's frame, which means that the option for a strategy is a function of the problem's specificity. This is how humans react, in an implicit or explicit way, depending on the problem's evolution.

In some sense these two forms of human behaviour can be emulated by artificial entities, the software agents.

What is an agent? While there is no generally agreed definition of what an 'agent' is [4], the term is usually used to describe self-contained programs, which can control their own actions, based on their perceptions of their operating environment. The aim of agent design is to create programs which interact intelligently with their environment.

The software agents are entities that have the ability to plan, to establish their actions ahead of time, to develop appropriated problem's solving strategies, to communicate, or to share resources. In a reactive system agents have the possibility to follow events as they occurs in the environment. To be an effective task force, agents must also cooperate. Sharing results or sharing tasks, are two common ways of doing it. In result sharing, agents work on inter-related sub-problems, interpreting and sharing knowledge or data. Agents do not act singly. The results obtained by a specific one may be used by the others. In task sharing, the problem is fragmented into sub-problems and distributed by a group of agents. Task distribution is done according to a prior deal in

order to decide the allocation of tasks. Once task allocation is done, each agent work per se.

Agents typically have the following properties:

- autonomy – agents operate without others having direct control of their actions and internal state;
- social ability – agents interact with other agents (and possibly humans) through some kind of language (message passing, ...);
- reactivity – agents are able to perceive their environment (which may be the physical world, a virtual world of electronic networks, or a simulated world including other agents) and respond to it;
- goal-oriented – an agent does not simply act in response to the environment;
- proactivity – as well as reacting to their environment, agents are also able to take the initiative, engaging in goal-directed behaviour.

Multi-Agent Systems refers to the algorithmic solutions of problems dealing with agents; how agents should interact, avoid conflicts or organise cooperative behaviour in order to fulfil common goals.

Agents are active objects in a Multi-Agent system. They can represent different degrees of heterogeneity. At a low heterogeneity level, agents are nearly identical and may differ only in the resources available. At a medium level, agents have different problem solving methods and expertise. If the only common property is the interaction language used, while other characteristics are quite different, agents are highly heterogeneous.

Gasser [5] identified the following problems as being inherent for all Multi-Agent Systems research:

- Description, decomposition, distribution, and allocation of tasks among agents;
- Communication and interaction of agents (how, what and when);
- Coordination and coherence in decision making and acting of agents;
- Representation and reasoning about the actions, plans, and knowledge of other agents;
- Recognition and reconciliation of disparate viewpoints and conflicting intentions among agents.

A natural way of programming agents is to use an Object Oriented programming language. In this context, objects are program structures which hold both data and procedures for operating on those data. In object oriented programming, the data are stored in slots within the object and the procedures are called methods. In most object oriented languages, objects are created from templates called classes which specify the composition of the object, the data it can hold and the methods it uses. All the objects derived from the same class are similar in terms of the methods and slots they possess, although the data values of different objects may vary. The classes themselves are arranged in a hierarchy, with subordinate classes inheriting the methods and slots of superior classes, but adding additional ones or replacing the superior one's slot and methods with more specialised substitutes.

Classes can represent the agents in a system. Once a set of classes has been defined, creating instances from them generates individual agents. The advantage of the object-oriented approach is that the slots can represent the internal states of the agent (including its working memory and the rules), while the methods can implement the rule interpreter. In addition, the object oriented approach leads naturally to a useful encapsulation, with each agent clearly distinguishable within the program. The fit between object orientation and Multi-agent Modelling is so close that nearly all Multi-agent Simulation systems are written using Object Oriented programming languages. Examples of used languages are C++, Objective C, Smaltalk and Java.

Multi-Agent Simulation Architecture for Assembly Lines

We have been working on the specification of a multilevel architecture for the simulation of Assembly Lines, based on the concept of Agents and Multi-Agent systems, that can after be implemented on a centralised or distributed environment. This architecture will be implemented according to the principles of Object Oriented Modelling.

Our objective is to build an architecture that overcomes the SimBa limitations, so the architecture must support the simulation of multi-model and mixed-model lines, the simultaneous simulation of several lines, the simulation of the transient models period, and the distribution of the human resources according to their skills.

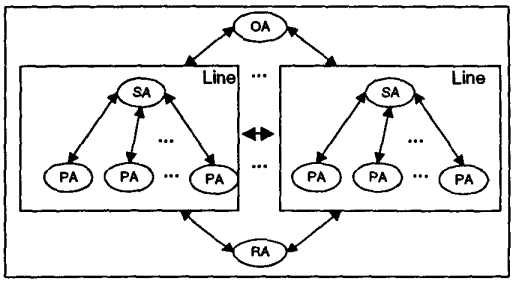

Fig.2 Specification of the Multi-Agent Architecture.

In the architecture four different kind of agents cooperate to reach a common goal: the manufacturing of the products assigned to the line (or the lines, because it is possible to simulate more than one line at the same time) according to a specified Cycle Time or Number of Workstations. Figure 2 illustrates the architecture. Although it is not very explicit in the Figure, all the agents in the system can communicate with all the others.

The four types are:

- Supervisor Agents (SA) – agents that simulate the responsible for the operation of an Assembly Line. There must be at least one SA for each simulated line. These agents are the ones that detain all the knowledge about the work that must be done in the line, according to what Cycle Time, and decide how many workstations (PA agents) will be needed and which operations should be done in each one. These type of agents are capable of creating Production Agents;
- Production Agents (PA) – agents that simulate the workstations of a line. They receive the orders of the SA that creates them and execute the operations assigned to them in the time specified;

- Resources Agent (RA) – agents that manage the resources that must be employed. This type of agent will be constituted of sub-agents, organised in a hierarchical structure, that are responsible for the different resources that can be used, such as tools, equipment, and operators;
- Observer Agent (OA) – an agent that observes the simulation run, gather and analyse the information about the system performance. The OA can detect some inefficiency in the line and suggest corrective actions to the SA responsible for that line.

The SA agents decide how many PA agents will be needed to operate the line and are the responsible for their creation. The SA distributes the operations that must be done among the PA agents and tells them how many time can be spent to process each product in each workstation (the time each PA can spend to work a product), and also tells them how many equal products should be done.

Each PA requests the resources needed to perform the operation, equipment and operators, by dialoguing with the RA agents, in order to respect the time indicated by the SA. If it is not possible to respect the Cycle Time the PA should inform the SA that he couldn't do the job in the specified time. Them the SA is responsible to change the distribution of operations among the PA's, create another PA or just let the PA disrespect the Cycle Time. When the production of one product ends, the PA must generate an event to notify the PA of the next workstation of the arrival of another product. Each time a product is ended in a workstation, the related PA notifies the OA of that event. When a PA ends the processing of all the products it informs the SA and also the OA. By that time the SA can assign another work to it, which means that it is possible to simulate the introduction of another product in the line, or just let it disappear.

After a set of products is completely done the OA can build the final report of the simulation of the manufacturing of that product. From time to time (which will depend on the type of system being simulated and the operations duration) and whenever requested by the user the OA can also show the information gathered by that time. The OA communicates with the SA in order to inform them of inefficiencies of the lines operations. The SA is then responsible for implementing corrective actions suggested by the OA or other actions he thinks useful to solve the problem.

The SA agents present in the system can communicate, and also the PA from different lines. It is possible that a PA "subcontract" another PA work, or resources from a PA in another line (supervised by another SA).

With the architecture specified it is possible to overcome the SimBa system limitations. It becomes possible to simulate the operation of more than one line at the same time (it means that there must be more than one SA). The transient periods (introduction of another model in the line) are simulated, and the cooperation between the PA and RA improves the distribution of human resources.

The possibility of implementing this Multi-Agent architecture in the Swarm Toolkit [2] [7] is currently being studied. Swarm is a Multi-Agent software platform for the simulation of complex systems. It is a fully object-oriented, dynamically scheduled, discrete event simulation package. Swarm is being developed at the Santa Fe Institute.

The goal in designing Swarm was to ease the process of simulating large numbers of interacting agents. By providing access to appropriate user-interface, simulation management, and analysis tools, Swarm enables researchers to concentrate on their research problems. It is only necessary to write code describing the details of the specific problems instead of having to develop an entire simulation from scratch.

Some of the most important features of Swarm are [7]:

- Swarm is a general-purpose package for simulating concurrent, distributed artificial worlds. Its core is an object oriented framework for defining the behaviour of agents and other objects that interact during a simulation. An agent is any actor in a system, any entity that can generate events that affect itself and other objects.
- Swarm uses the individual based modelling approach. This allows each agent to have its own set of internal state variables affected by its past experiences and determining its next actions.
- The fundamental component that organises agents of a Swarm model is an object called a *swarm*. A *swarm* is a collection of agents with a schedule of events over those agents. In addition to being containers for agents, *swarms* can themselves be agents. Hierarchical models can be built by nesting multiple *swarms*. Swarms can be nested to directly represent multi-level simulations, and they can be used by the agents themselves as models of their own world.
- A *swarm* is not just a collection of objects but also a schedule. Each *swarm* maintains its own schedule independent of other *swarms*, and all agents in the same *swarm* adhere to the same schedule. A nested structure of *swarms* is thus a nested hierarchy of schedules. Actions are scheduled with reference to a global clock although during execution, different *swarms* are allowed to slip from global synchrony.
- An action is a triplet consisting of a message to be sent, an agent, or a collection of agents, to send the message to, and a time to send the message. During execution, the kernel scheduler traverses the schedule list to inform agents to act by sending the message to them at the specified time. Agents that receive a message perform actions corresponding to their action models. The action may generate another state, which is inserted in the schedule list. In this way, Swarm works as discrete event simulator. Swarm also allows users to design agents with fixed schedules that continue to repeat the same actions, which works as a time stepped simulator.
- Swarm supports both interactive simulation and batch simulation.
- Swarm provides the "probe" facility. Using probes, the state of any object can be read or set and any method can be called in a generic fashion. No extra code is required to use these features. Probes are used to make data analysis tools work in a general way and are also the basis of graphical tools to inspect objects in a running system.

The agents specified in the proposed architecture must be implemented and organised in *swarms* in order to be possible the utilisation of the Swarm Toolkit.

The generic organisation of *swarms* and *sub-swarms* will be:

- Each line will be implemented as a *swarm*, containing all the PA and the SA of the line, and the scheduling of all the events that affects them.
- The hierarchy of agents that are responsible for the resources (RA) will also be organised in another *swarm*.
- The OA will be implemented with the aid of the "probe" facility, and so the efforts in implementing this kind of agents will be focused in analysing and controlling the manufacturing lines.

There must also be a *swarm* that contains all the *sub-swarms* referred, in order to simulate the interactions between all the lines and all the agents of the manufacturing system simulation.

Conclusions

The use of an information system that gives the user different line configurations and qualitative and quantitative information about the line performance can be an helpful framework to the managers of the production lines. Especially in systems where the need to change line configurations is frequent the analysis through simulation of the transient interval (non steady state conditions) can help making some adjustments in order to minimise the associated effects which cant result in lower performance of the line. The development of a system of this type, called SimBa, to a real manufacturing system was the first approach to solve this problem by means of simulation. However SimBa system has some limitations.

In this paper we proposed a Multi-Agent architecture for Line Balancing and Simulation that overcomes the SimBa limitations.

Although the proposed architecture seems more flexible than the developed system SimBa, capable of simulate multi-model and mixed-model lines, simulate the transient periods (introduction of another product in the line), simulate the operation of more than one line and improve the distribution of human resources according to their skills, there is still a lot of work to be done.

The implementation of the architecture in the Swarm Toolkit seems very promising and we are working on it.

REFERENCES

[1] Boctor, Fayez F. 1995. "A Multiple-rule Heuristic for Assembly Line Balancing." *Journal of the Operational Research Society.*

[2] Bruhn, Peter. 1997. "Enterprise Simulation using Multi-Agent Systems Modelling and the Swarm Toolkit". MSc Thesis, Graduate College of the University of Illinois at Urbana-Champaign.

[3] Fishwick, Paul A. 1992. "Simpack: Getting Started width Simulation Programming In C And C++". Dept. of Computer & Information Science, University of Florida.

[4] Franklin, S. and Graesser, A. 1996. "Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents". In *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages,* Springer-Verlag.

[5] Gasser, L. 1991. "Social conceptions of knowledge and action: DAI foundations and open systems semantics". *Artificial Intelligence 47*, pag. 107-138.

[6] Law, Averill M., and W. David Kelton. 1982. *Simulation Modelling and Analysis.* McGraw-Hill.

[7] Minar, Nelson, et. al. 1996. "The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations". Http://www.santafe.edu/projects/swarm/.

[8] Praça, Isabel. 1996. *Balanceamento e Simulação de Linhas de Fabrico Manuais.* MSc Thesis, Faculty of Engineering, University of Porto, Portugal (in Portuguese).

[9] Praça, Isabel. 1997. "Simulation of Manual flow Lines". In *Proceedings of the ESS97-9th European Simulation Symposium*, 19 a 23 October, Passau, Germany, pag. 337-341.

[10] Rumbaugh, James. 1991. *Object-Oriented Modelling and Design.* Prentice-Hall International.

[11] Wild, Ray. 1989. *Mass-Production Management.* John Wiley & Sons.